**INDEX**

|  |  |
| --- | --- |
| **S/N** | **Experiment Name** |
| 01 | Amplitude modulation and demodulation |
| 02 | DSB-SC modulator and detector |
| 03 | SSB-SC modulator and detector |
| 04 | Time division multiplexing and de-multiplexing |
| 05 | Verification of sampling theorem |
| 06 | Pulse amplitude modulation and demodulation |
| 07 | Pulse width modulation and demodulation |
| 08 | Pulse Position modulation and demodulation |

**Experiment No - 1:**

**Experiment Name:** Amplitude modulation and demodulation.

**Objectives:**

**1.** To understand modulation and demodulation process of amplitude.

**2.** To observe the change of amplitude and frequency.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

Modulation is defined as the process by which some characteristics of a carrier signal is varied in accordance with a modulating signal. The base band signal is referred to as the modulating signal and the output of the modulation process is called as the modulation signal. Amplitude Modulation is a process by which amplitude of the carrier signal is varied in accordance with the instantaneous value of the modulating signal, but frequency and phase of carrier wave remains constant. The envelope of the modulating wave has the same shape as the base band signal provided the following two requirements are satisfied.

1. The carrier frequency fc must be much greater than the highest frequency components fm of the message signal m (t) i.e. fc >>fm

2. The modulation index must be less than unity. if the modulation index is greater than unity, the carrier wave becomes over modulated.

The modulating and carrier signal are given by

Vm(t) = Vm sinWmt

VC(t) = VC sinWCt

The modulation index is given by, ma = Vm/ VC.

Vm = Vmax – Vmin and

VC = Vmax + Vmin

The amplitude of the modulated signal is given by,

VAM(t) = VC (1+ma sinWmt) sinWCt

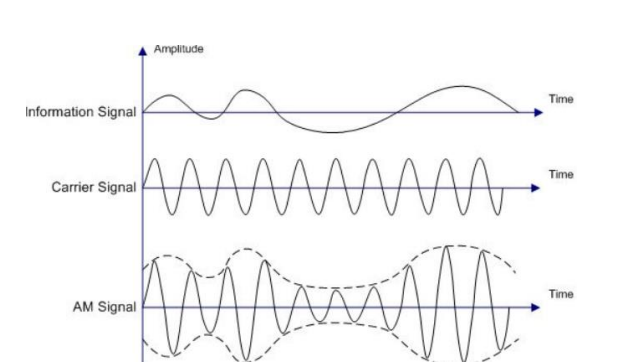
Where,

Vm = maximum amplitude of modulating signal

VC = maximum amplitude of carrier signal

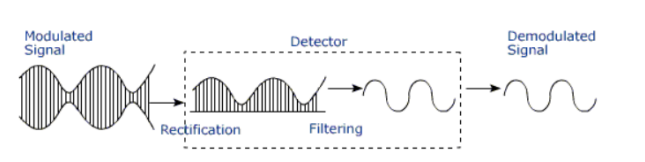
Vmax = maximum variation of AM signal

Vmin = minimum variation of AM signa



**Fig:** AM Modulation

The process of extracting the information bearing signal from the modulated bandpass signal is known as demodulation or detection. To recover the low frequency baseband signal, the received signal is first rectified and then filtered as depicted in Figure 2.



**Fig:** AM Demodulation

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

Am = input("Enter amplitude of the message signal: ");

Fm = input("Enter frequency of the message signal: ");

Ac = input("Enter amplitude of the carrier signal: ");

Fc = input("Enter frequency of the carrier signal: ");

Fs = input("Enter sampling frequency: ");

t = 0:0.001:1;%defining the time range

MI = Am/Ac; %defining modulating index

Mt =Am.\*sin(2\*pi\*Fm\*t);%defining the message signal

Ct =Ac.\*sin(2\*pi\*Fc\*t);%defining the carrier signal wave

St =(1+MI\*Mt).\*Ct;%Amplitude Modulated wave, according to the standard definition

O = St;

%defining a loop for rectification of the modulated signal

for i=1:length(t)

if O(i)<=0

O(i)=0;

end

end

[x,n] = butter(2,2\*pi\*Fm/Fs);%defining butterworth lpf of 2nd order

O1 = filter(x,n,O);

O2 = filter(x,n,O1);

O3 = filter(x,n,O2);

subplot(5,1,1)%plotting the message signal wave

plot(t,Mt);

title("Message Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,2);%plotting the carrier signal wave

plot(t,Ct);

title("Carrier Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,3);%plotting the amplitude modulated wave

plot(t,St);

title("Modulated AM Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,4);%plotting the Rectified modulated wave

plot(t,O);

title("Rectified modulated Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,5);%plotting the demodulated wave

plot(t,O3);

title("Demodulated Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

**Command Window:**

Enter amplitude of the message signal: 1

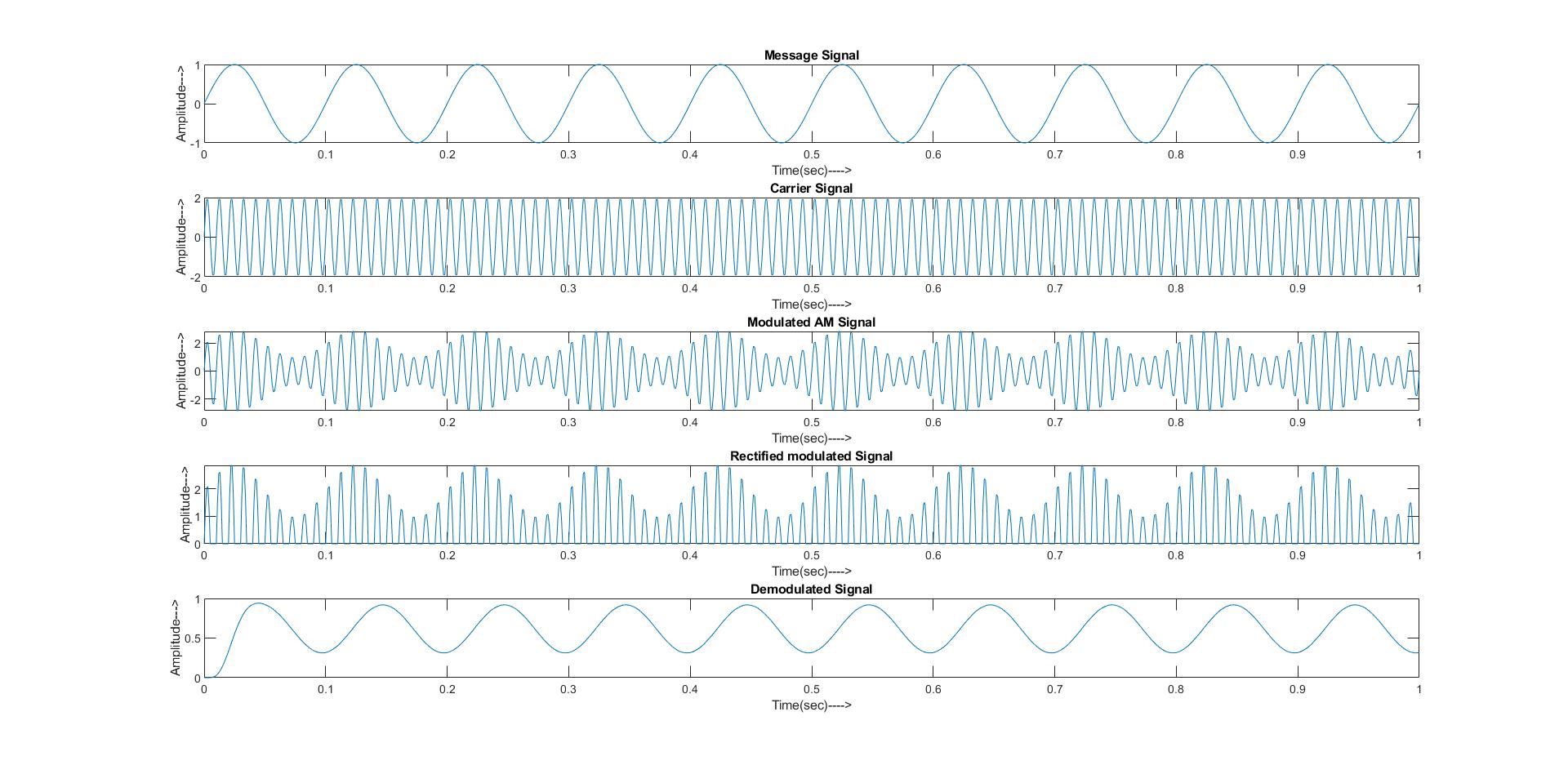
Enter frequency of the message signal: 10

Enter amplitude of the carrier signal: 2

Enter frequency of the carrier signal: 100

Enter sampling frequency: 1000

**Output:**



**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 2:**

**Experiment Name:** DSB-SC modulator and detector.

**Objectives:**

**1.** Demonstrate the modulation and demodulation process of DSB-SC.

**2.** realize the real-life difficulties and challenges in designing coherent demodulators.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

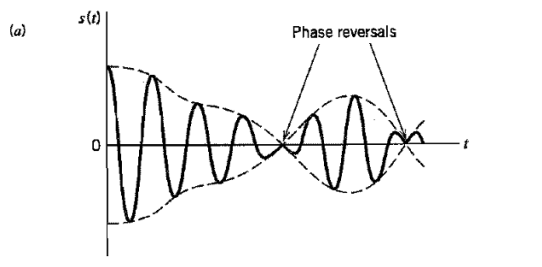
**Theory:**

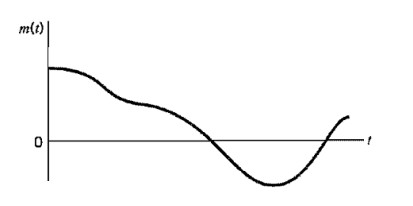
Double Side Band Suppress Carrier (DSB-SC) is one type of Amplitude Modulation. The modulation process is straightforward: the message is multiplied by a high-frequency carrier. The modulated signal occupies double the bandwidth of the baseband signal.

Consider two sinusoids, or sinusoids, AmCos(2πfmt) (Message Signal) and AcCos(2πfc t) (Carrier Signal ).

Recovering the message signal from the demodulated signal is performed coherently. That is, the demodulated signal is multiplied by a high-frequency sinusoid in perfect synchronization (in phase and frequency) with the incoming carrier.

This requirement poses a challenge on the design of the demodulator circuit, as it would then require a part for carrier-recovery. Failing to accomplish perfect synchronization will result in phase mismatch or frequency mismatch, leading to some form of distortion in the recovered signal.





**Fig:** DSB-SC Technique

Multiplying the modulated signal with a local carrier will produce a baseband signal as well as a signal modulated at double the carrier frequency. Therefore, an LPF is needed at the far end of the demodulator to recover the baseband signal.

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

Fm = input("Enter frequency of the message signal: ");

Fc = input("Enter frequency of the carrier signal: ");

Fs = input("Enter sampling frequency: ");

T = input("Enter duration over which signal to be plotted: ");

C = input("Enter value of the capacitor of the filter: ");

t = 0:T/Fs:T;%defining the time range

Mt =cos(2\*pi\*Fm\*t);%defining the message signal

Ct =cos(2\*pi\*Fc\*t);%defining the carrier signal wave

St = Mt.\*Ct;%DSB-SC Modulated wave, according to the standard definition

O = St.\*Ct;

R = 1/(2\*pi\*Fm\*C);%defining reactance of the capacitor

H = (1/(R\*C))\*exp(-t/(R\*C));

h = conv(H,conv(O,H));

t1 =t;

for i=length(t)+1:length(h)

t1(i)=0;

end

subplot(4,1,1)%plotting the message signal wave

plot(t,Mt);

title("Message Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(4,1,2);%plotting the carrier signal wave

plot(t,Ct);

title("Carrier Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(4,1,3);%plotting the modulated wave

plot(t,St);

title("DSB-SC Modulated Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(4,1,4);%plotting the demodulated wave

plot(t1,h);

title("Demodulated Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

**Command Window:**

Enter frequency of the message signal: 10

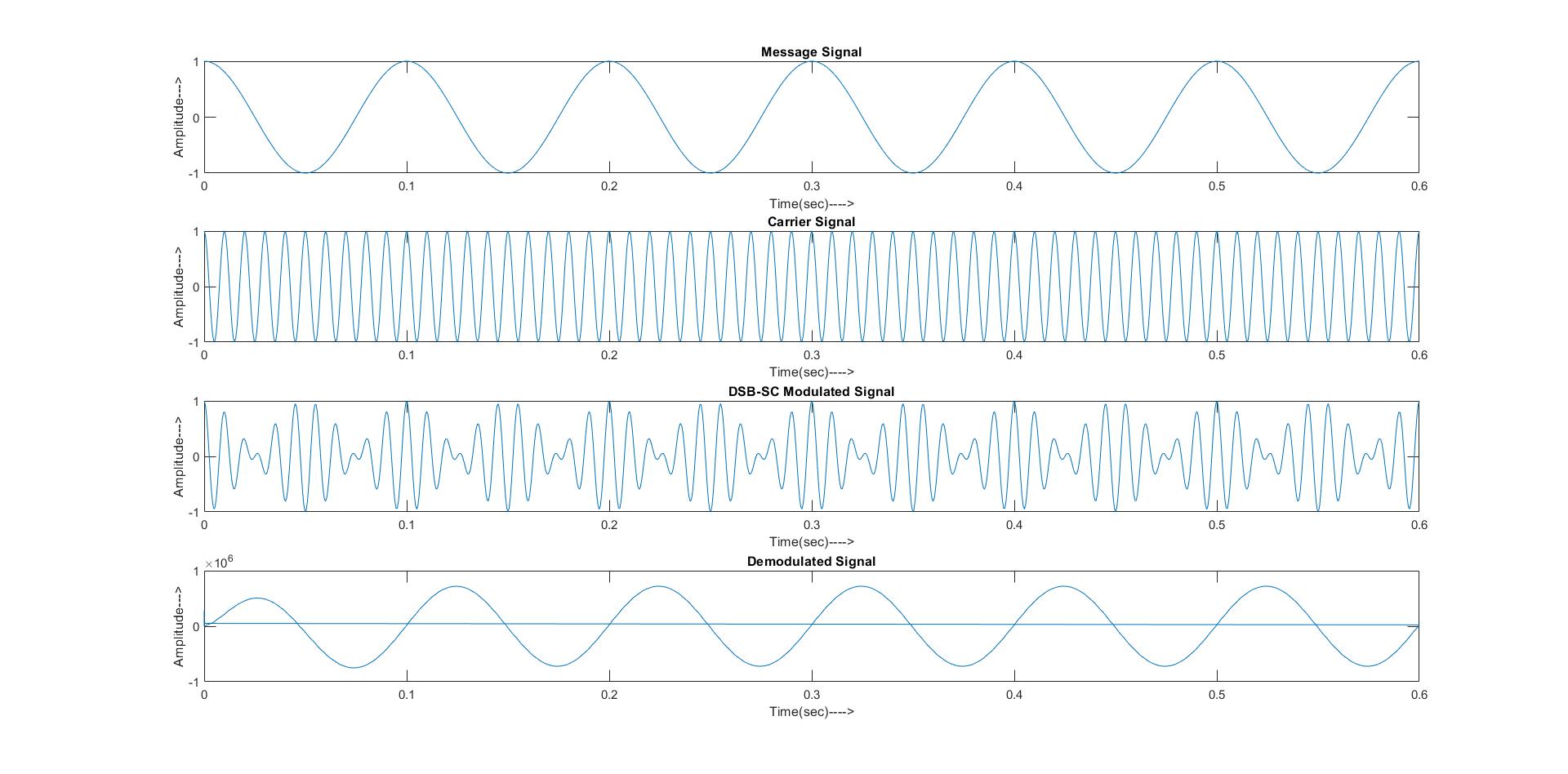
Enter frequency of the carrier signal: 100

Enter sampling frequency: 1000

Enter duration over which signal to be plotted: .6

Enter value of the capacitor of the filter: 1e-8

**Output**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 3:**

**Experiment Name:** SSB-SC modulator and detector.

**Objectives:**

**1.** Demonstrate the modulation and demodulation process of SSB-SC.

**2.** Realizing the real-life difficulties and challenges in designing coherent demodulators.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

Single sideband (SSB) is a common analog modulation scheme for voice communications. With SSB only one sideband (either the upper USB or the lower LSB) is present in the modulated carrier. That is acceptable because the two sidebands contain the same information, so the elimination of one sideband does not cause a loss of information.

SSB uses radio spectrum efficiently: for a given message signal, only half as much bandwidth is occupied by the modulated carrier (compared with DSB or AM). SSB is used for amateur (ham) radio, citizens’ band (CB) radio, and short-wave broadcasting. There is more than one way to generate SSB carriers. One method is to use a DSB modulator and then eliminate one sideband (either the lower or the upper) with a filter. That method is conceptually simple but has a significant drawback.

The filter can be challenging to design: it must have a quite sharp roll-off that will pass the one sideband but reject the other sideband that is just the other side of the carrier frequency. In the present experiments SSB carriers will be generated by a different method. The method employed here is known as the phasing method, and it incorporates a Hilbert transform. Hilbert Transform In general, a signal m(t) has a Hilbert transform, in a Hilbert transform, both the input and the output are in the time domain. This is unlike the Fourier transform, for which the input is in the time domain and the output is the frequency domain description of the input. The Hilbert transform is a linear, time-invariant system. If the input is a sinusoid, the output is also a sinusoid of the same frequency. For a sinusoidal input, the output has a phase that is less than that of the input by radians.

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

Fm = input("Enter frequency of the message signal: ");

Fc = input("Enter frequency of the carrier signal: ");

Fs = input("Enter sampling frequency: ");

%defining the time range

t = 0:0.001:0.4;

Mt =cos(2\*pi\*Fm\*t);%defining the message signal

Ct =cos(2\*pi\*Fc\*t);%defining the carrier signal wave

DSB1 = Mt.\*Ct;%DSB-SC Modulated wave

M1 = cos(2\*pi\*Fm\*t - (pi/2));

N1 = cos(2\*pi\*Fc\*t - (pi/2));

DSB2 = M1.\*N1;

USB = DSB1-DSB2;%Generating upper sideband signal

LSB = DSB1+DSB2;%Generating lower sideband signal

USBMult = USB.\*Ct;

%defining butterworth filter

[x,y] = butter(2,(2\*pi\*Fm)/Fs);

F1 = filter(x,y,USBMult);

F2 = filter(x,y,F1);

F3 = filter(x,y,F2);

F4 = filter(x,y,F3);

subplot(5,1,1);%plotting the message signal wave

plot(t,Mt,'k',t,M1,'--b');

title("Baseband signal and its hilbert transform");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,2)%plotting the Carier signal wave

plot(t,Ct,'k',t,N1,'--b');

title("Carrier signal and its hilbert transform");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,3);%plotting Upper side band signal

plot(t,USB);

title("Upper Sideband Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,4);%plotting lower sidea band signal

plot(t,LSB);

title("Lower Sideband Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(5,1,5);%plotting the message signal wave

plot(t,F4);

title("Demodulated Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

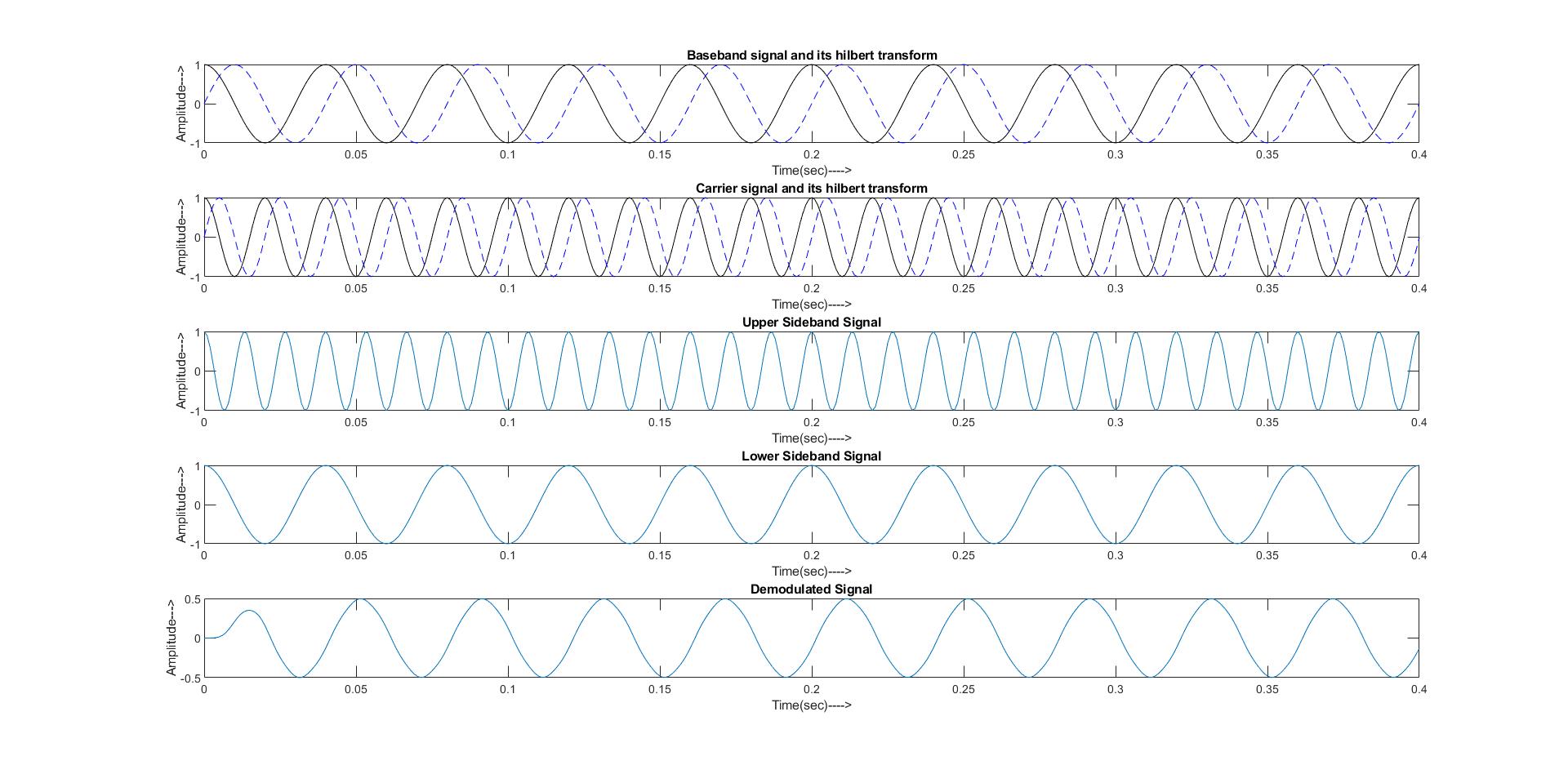
**Command Window:**

Enter frequency of the message signal: 25

Enter frequency of the carrier signal: 50

Enter sampling frequency: 1000

**Output:**



**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 4:**

**Experiment Name:** Time division multiplexing and demultiplexing.

**Objectives:**

**1.** To understand time division multiplexing and demultiplexing.

**2.** To implement a time division multiplexing system using MATLAB.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

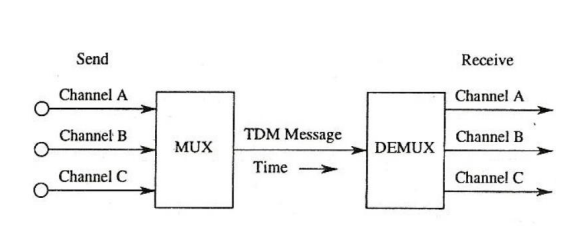
MATLAB version: R2020a

**Theory:**

The TDM is used for transmitting several analog message signals over a Communication channel by dividing the time frame into slots, one slot for each message signal. The four input signals, all band limited by the input filters are sequentially sampled, the output of which is a PAM waveform containing samples of the input signals periodically interlaced in time. The samples from adjacent input message channels are separated by Ts/M, where M is the number of input channels. A set of M pulses consisting of one sample from each of the input M-input channels is called a frame.

At the receiver the samples from individual channels are separated by carefully synchronizing and are critical part TDM. The samples from each channel are filtered to reproduce the original message signal. There are two levels of synchronization. Frame synchronization is necessary to establish when each group of samples begins and word synchronization is necessary to properly separate the samples within each frame.

Besides the space diversity & frequency diversity there is a method of sending multiple analog signals on a channel using “TIME DIVISION MULTIPLEXING & DEMULTIPLEXING” Technique.



**Fig**: Multiplexing and Demultiplexing

**Code:**

clear all; %clear workspace window

close all; %close all window except command window

clc; %clear command window

N = input("Enter the number of sinals to be multiplexed: ");

f = zeros(1,N);

for i = 1:N

f(i) = input("Enter the frequency of the singal: ");

end

fs = 4\*max(f);

T = input("Enter the duration over which the signal to be plotted: ");

t = 0:T/fs:T;

Q = zeros(1, N\*length(t));

R = Q;

S = Q;

for i=1:N

Q(:,((i-1)\*length(t))+1:i\*length(t)) = cos(2\*pi\*f(i)\*t);

end

j = 1;

for i=0:N:length(Q)-N

for n = 1:N

R(i+n) = Q(j+(n-1)\*length(t));

S(j+(n-1)\*length(t))=R(i+n);

end

j = j +1;

end

t1 = 0:T/fs:T;

for i = length(t1)+1:length(Q)

t1(i)=t1(length(t1));

end

subplot(N+2,1,1)%plotting Carrier Signals

plot(t1,Q)

title("Signals to be multiplexed");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

subplot(N+2,1,2)%plotting Multiplexed Signals

stem(t1,R)

title("Multiplexed Sinals");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

for i = 1:N

subplot(N+2,1,i+2)%plotting De-Multiplexed Signals

plot(t,S(:,((i-1)\*length(t))+1:i\*length(t)));

title("Demultiplexed Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

end

**Command Window:**

Enter the number of signals to be multiplexed: 2

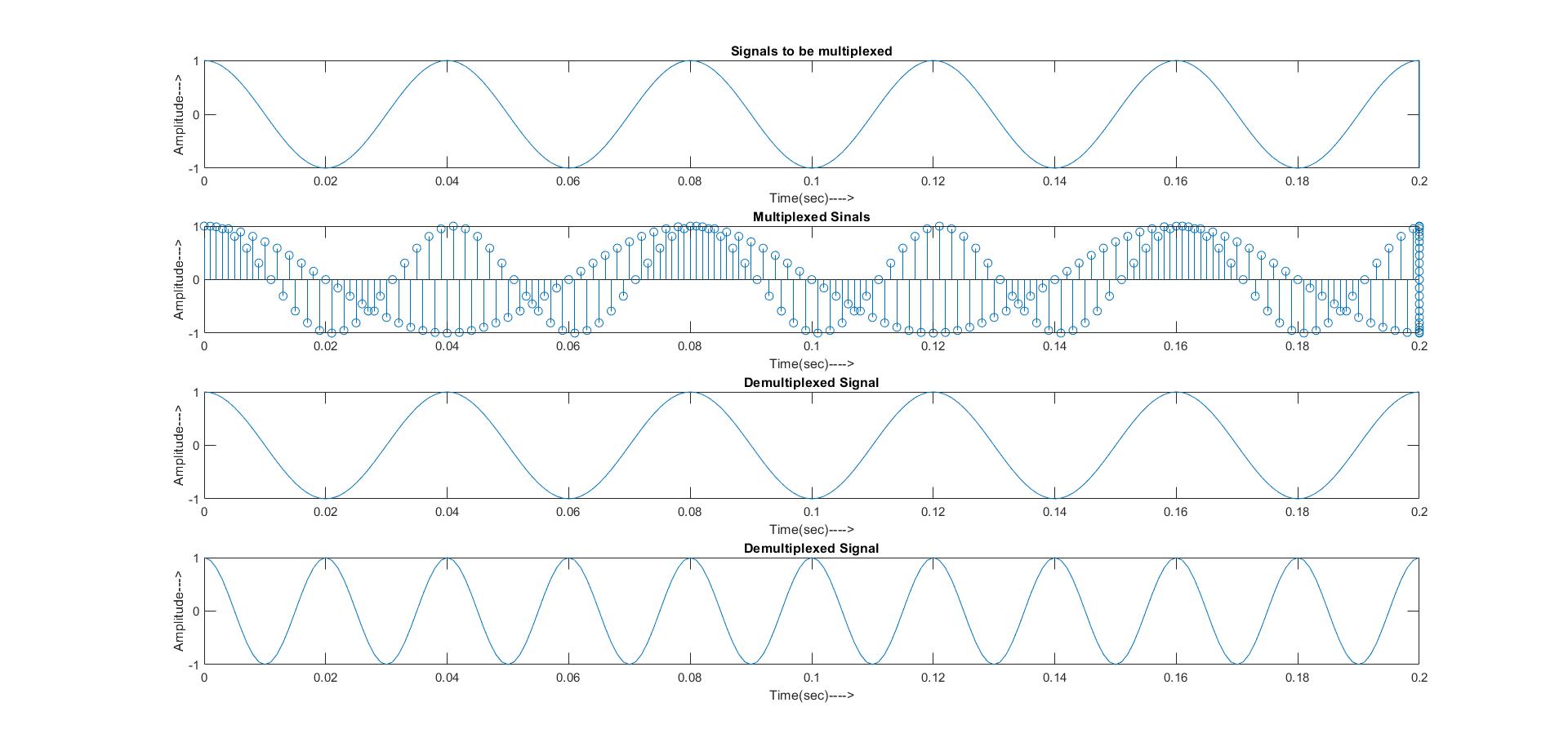
Enter the frequency of the signal: 25

Enter the frequency of the signal: 50

Enter the duration over which the signal to be plotted: .2

\

**Output:**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 5:**

**Experiment Name:** Verification of sampling theorem.

**Objectives:**

**1.** To study the sampling theorem and its reconstruction.

**2.** To study the effect of amplitude and frequency variation of modulating signal on the output

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

A band limited signal of finite energy which has no frequency components higher than fm Hz, is completely described by specifying the values of the signal at instants of time separated by ½ fm seconds.

The sampling theorem states that, if the sampling rate in any pulse modulation system exceeds twice the maximum signal frequency, the original signal can be reconstructed in the receiver with minimum distortion.

Fs > 2fm is called Nyquist rate.

Where fs – sampling frequency

Fm – Modulation signal frequency.

If we reduce the sampling frequency fs less than fm, the side bands and the information signal will overlap and we cannot recover the information signal simply by low pass filter. This phenomenon is called fold over distortion or aliasing. There are two methods of sampling. (1) Natural sampling (2) Flat top sampling.

Sample & Hold circuit holds the sample value until the next sample is taken. Sample & Hold technique is used to maintain reasonable pulse energy. The duty cycle of a signal is defined as the ratio of Pulse duration to the Pulse repetition period. The duty cycle of 50% is desirous taking the efficiency into account

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

t = 0:0.001:0.1;

t1 = zeros(1,length(t));

f = input("Enter the baserband signal frequency: ");

x = sin(2\*pi\*f\*t);

n = input("Enter the integer which decides the sampling frequency: ");

for i = 1:length(t)

if n\*i<=length(t)

t1(n\*i)=1;

end

end

s1 = x.\*t1;

[den,num] = butter(1,2\*pi\*f/1000);

s11 = filter(den,num,s1);

subplot(2,1,1)

stem(t,s1)

title("Sampling rate less than Nyquist rate");

subplot(2,1,2)

plot(t,s11)

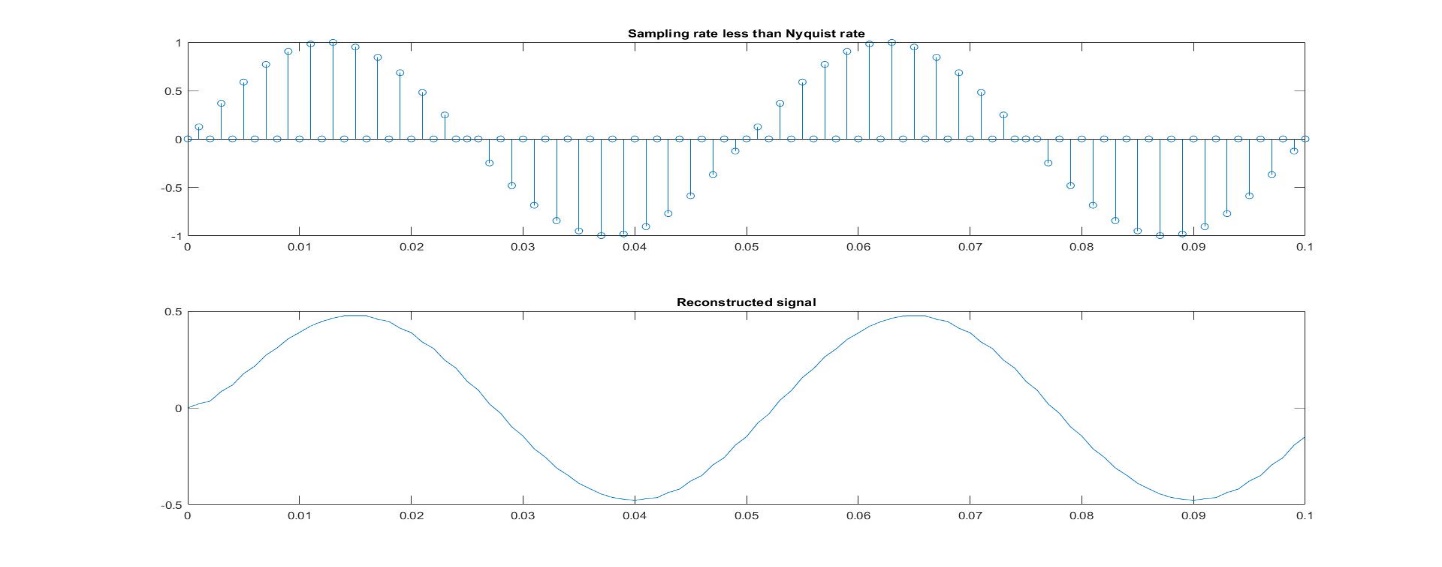
title("Reconstructed signal");

**Command Window:**

Enter the baseband signal frequency: 20

Enter the integer which decides the sampling frequency: 2

**Output:**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 6:**

**Experiment Name:** Pulse Amplitude Modulation and Demodulation.

**Objectives:**

**1.** To study the Pulse amplitude modulation & demodulation Techniques.

**2.** To study the effect of amplitude and frequency variation of modulating signal on the output.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

Pulse modulation is used to transmit analog information. In this system continuous wave forms are sampled at regular intervals. Information regarding the signal is transmitted only at the sampling times together with syncing signals.

At the receiving end, the original waveforms may be reconstituted from the information regarding the samples.

The pulse amplitude modulation is the simplest form of the pulse modulation. PAM is a pulse modulation system is which the signal is sampled at regular intervals, and each sample is made proportional to the amplitude of the signal at the instant of sampling. The pulses are then sent by either wire or cables are used to modulated carrier.

The two types of PAM are i) Double polarity PAM, and ii) the single polarity PAM, in which fixed dc level is added to the signal to ensure that the pulses are always positive. instantaneous PAM sampling occurs if the pulses used in the modulator are infinitely short.

Natural PAM sampling occurs when finite-width pulses are used in the modulator, but the tops of the pulses are forced to follow the modulating waveform. Flat-topped sampling is a system quite often used because of the ease of generating the modulated wave.

PAM signals are very rarely used for transmission purposes directly. The reason for this lies in the fact that the modulating information is contained in the amplitude factor of the pulses, which can be easily distorted during transmission by noise, crosstalk, other forms of distortion. They are used frequently as an intermediate step in other pulse-modulating methods, especially where time-division multiplexing is used.

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

t = 0:0.001:5; %1kHz sampling frequency for 1 sec

d = 0:1/5:5;

x = sin(2\*pi/4\*2\*t); %message signal

subplot(4,1,1)

plot(t,x);

title("Message Signal");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

y = pulstran(t,d,'rectpuls',0.1); %generating of pulse input

subplot(4,1,2)

plot(t,y);

title("Pulse Input");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

z = x.\*y; %PAM OUTPUT

subplot(4,1,3)

plot(t,z);

title("Pulse Input");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

[den,num] = butter(1,2\*pi\*0.5/1000);

s11 = filter(den,num,z);

s12 = filter(den,num,s11);

s13 = filter(den,num,s12);

subplot(4,1,4)

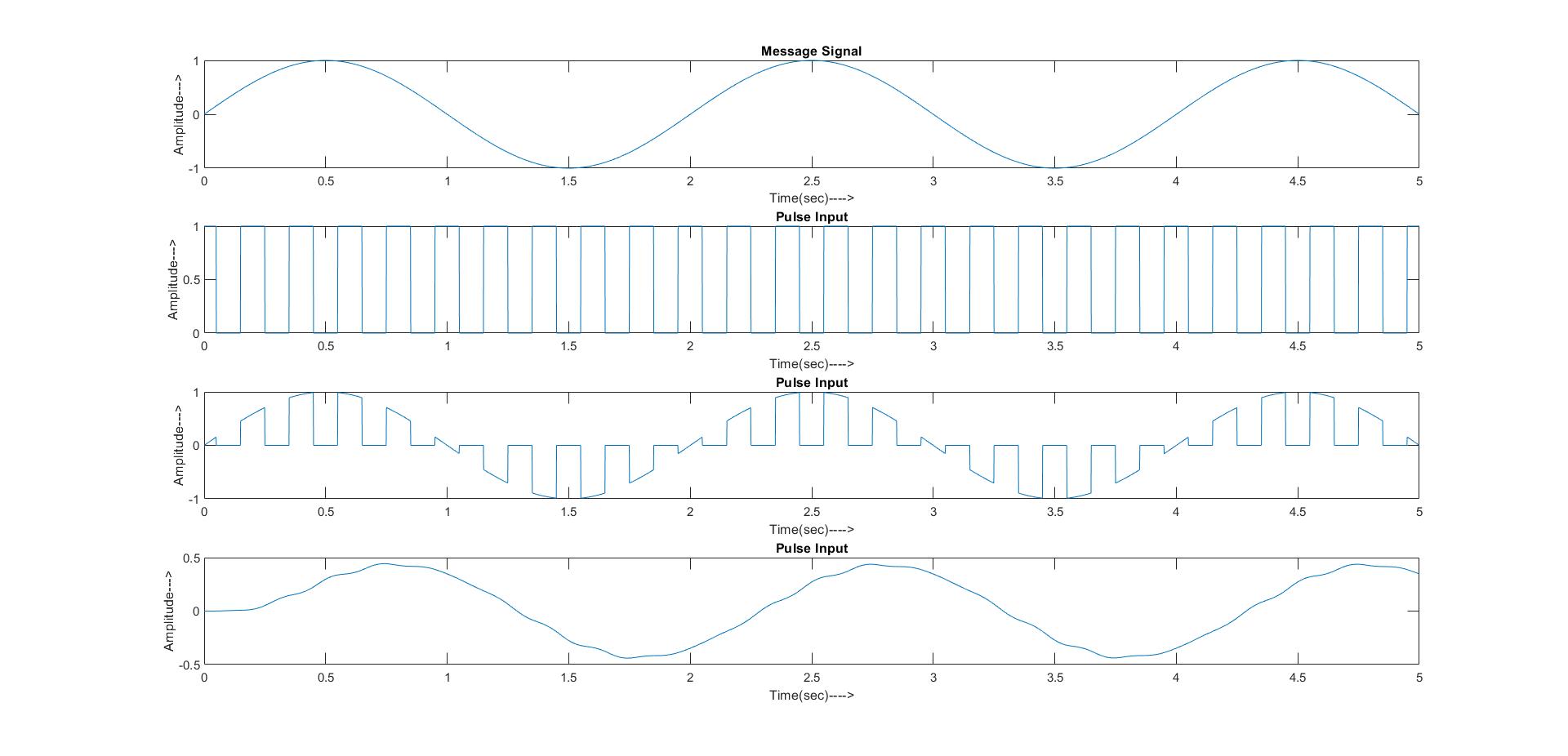
plot(t,s13);

title("Pulse Input");

xlabel("Time(sec)---->");

ylabel("Amplitude--->");

**Output:**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 7:**

**Experiment Name:** Pulse Width modulation and demodulation.

**Objectives:**

**1.** To study the Pulse Width Modulation (PWM) and Demodulation Techniques.

**2.** To study the effect of Amplitude and Frequency of Modulating Signal on PWM output.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

Pulse modulation is used to transmit analog information. In this system continuous wave forms are sampled at regular intervals. Information regarding the signal is transmitted only at the sampling times together with synchronizing signals.

At the receiving end, the original waveforms may be reconstituted from the information regarding the samples.

The pulse Width Modulation of the PTM is also called as the Pulse Duration Modulation (PDM) & less often Pulse length Modulation (PLM).

In pulse Width Modulation method, we have fixed and starting time of each pulse, but the width of each pulse is made proportional to the amplitude of the signal at that instant.

This method converts amplitude varying message signal into a square wave with constant amplitude and frequency, but which changes duty cycle to correspond to the strength of the message signal.

Pulse-Width modulation has the disadvantage, that its pulses are of varying width and therefore of varying power content. This means that the transmitter must be powerful enough to handle the maximum-width pulses. But PWM still works if synchronization between transmitter and receiver fails, whereas pulse-position modulation does not.

Pulse-Width modulation may be generated by applying trigger pulses to control the starting time of pulses from a mono stable multi-vibrator, and feeding in the signal to be sampled to control the duration of these pulses.

When the PWM signals arrive at its destination, the recovery circuit used to decode the original signal is a sample integrator (LPF).

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

fc = 1000;

fs = 10000;

f1 = 200;

t = 0.1/fs:((2\*f1)-(1/fs));

x1 = 0.4\*cos(2\*pi\*f1\*t)+0.5;

y1 = modulate(x1,fc,fs,'pwm');

subplot(4,1,1)

plot(t,x1);

axis([0 100 0 1]);

title('Modulating signal');

subplot(4,1,2)

plot(t,y1);

axis([0 1000 0 1]);

title("PWM");

x1\_recov = demod(y1,fc,fs,'pwm');

[den,num] = butter(1,2\*pi\*f1/fs);

s11 = filter(den,num,x1\_recov);

s12 = filter(den,num,s11);

subplot(4,1,3)

plot(t,x1\_recov);

%axis([0 100 0 1]);

title('Time domain recovered');

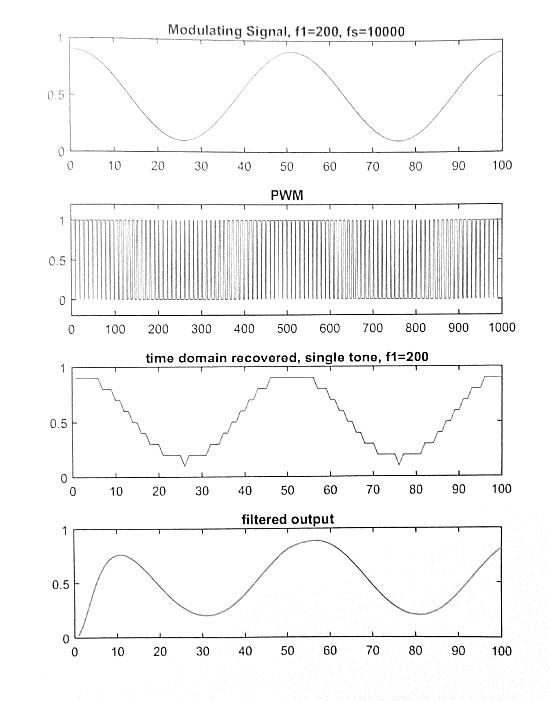
subplot(4,1,4)

plot(t,s12);

%axis([0 100 0 1]);

title('Filtered Output');

**Output:**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.

**Experiment No - 8:**

**Experiment Name:** Pulse Position modulation and demodulation.

**Objectives:**

**1.** To study the generation Pulse Position Modulation (PPM) and Demodulation.

**2.** To study the effect of Amplitude and the frequency of modulating signal on its output and observe the wave forms.

**Apparatus:**

1. PC with MATLAB installed:

Device name : Mahidur-PC

Processor : Intel(R) Core(TM) i5-8265U CPU @ 1.60GHz 1.80 GHz

Installed RAM: 8.00 GB (7.85 GB usable)

System type : 64-bit operating system, x64-based processor

MATLAB version: R2020a

**Theory:**

Pulse Modulation is used to transmit analog information in this system continuous wave forms are sampled at regular intervals. Information regarding the signal is transmitted only at the sampling times together with synchronizing signals.

At the receiving end, the original waveforms may be reconstituted from the information regarding the samples. Pulse modulation may be subdivided in to two types analog and digital. In analog the indication of sample amplitude is the nearest variable. In digital the information is a code.

The pulse position modulation is one of the methods of the pulse time modulation.PPM is generated by changing the position of a fixed time slot.

The amplitude& width of the pulses is kept constant, while the position of each pulse, in relation to the position of the recurrent reference pulse is valid by each instances sampled value of the modulating wave. Pulse position modulation into the category of analog communication. Pulse-Position modulation has the advantage of requiring constant transmitter power output, but the disadvantage of depending on transmitter receiver synchronization.

Pulse-position modulation may be obtained very simply from PWM. However, in PWM the locations of the leading edges are fixed, whereas those of the trailing edges are not. Their position depends on pulse width, which is determined by the signal amplitude at that instant. Thus, it may be said that the trailing edges of PWM pulses are, in fact, position-modulated. This has positive-going narrow pulses corresponding to leading edges and negative-going pulses corresponding to trailing edges. If the position corresponding to the trailing edge of an un modulated pulse is counted as zero displacement, then the other trailing edges will arrive earlier or later. They will therefore have a time displacement other than zero; this time displacement is proportional to the instantaneous value of the signal voltage. The differentiated pulses corresponding to the leading edges are removed with a diode clipper or rectifier, and the remaining pulses, is position-modulated.

**Code:**

clear all;%clear workspace window

close all;%close all window except command window

clc;%clear command window

fc = 100;

fs = 1000;

f1 = 80;

t = 0.1/fs:((2\*f1)-(1/fs));

x1 = 0.4\*cos(2\*pi\*f1\*t)+0.5;

y1 = modulate(x1,fc,fs,'ppm');

subplot(3,1,1)

plot(t,x1);

axis([0 15 0 1]);

title('Modulating signal');

subplot(3,1,2)

plot(t,y1);

axis([0 250 -0.2 1.2]);

title("PPM");

x1\_recov = demod(y1,fc,fs,'ppm');

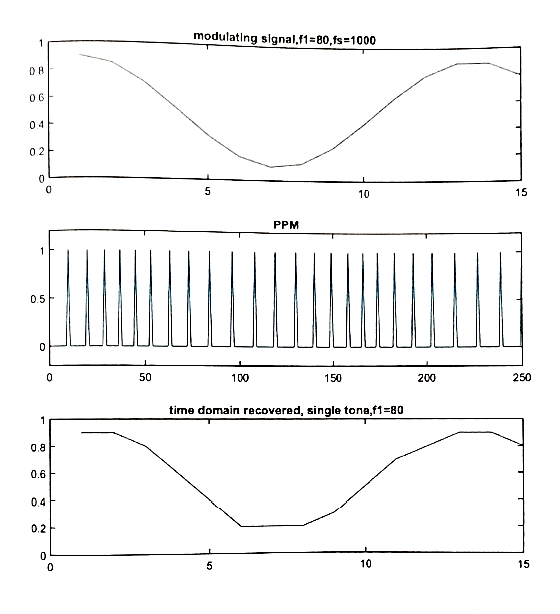
subplot(3,1,3)

plot(t,x1\_recov);

%axis([0 15 0 1]);

title('Time domain recovered');

**Output:**

****

**Discussion:** After doing the experiment, the experiment is completed successfully.